Amendments to the Claims:

This listing of claims will replace all prior versions and listings of claims in the application:

Listing of Claims:

1. (Currently Amended) A method for retrieving data comprising:

locking a linked list;

retrieving data from an element in the linked list and [also] advancing to a subsequent element while a breakpoint is not encountered;

marking the subsequent element in the linked-list as in-use [when] after encountering a breakpoint ~~is encountered~~;

creating a recommencement reference to the subsequent element; and

unlocking the linked list.

2. (Original) The method of claim 1 further comprising:

locking the linked list;

determining a subsequent element in the linked list according to the recommencement reference; and

retrieving data from the determined subsequent element.

3. (Original) The method of claim 1 wherein creating a recommencement reference to the subsequent element comprises:

retrieving a pointer to the subsequent element;

determining a process identifier for a current process; and

associating the pointer with the process identifier.

4. (Original) The method of claim 1 wherein marking the subsequent element in the linked-list as in-use comprises maintaining a count of the quantity of processes that require additional access to the element.

5. (Currently Amended) A method for deleting an element from a linked list comprising:

~~determining if the element to be deleted is in-use;~~

updating a recommencement reference to the element to refer to a data element ~~that is~~ subsequent to the data element to be deleted ~~when the~~ for each element [[in]] to be deleted which is determined to be [[is]] in-use; and

deleting the element.

6. (Original) The method of claim 5 wherein updating a recommencement reference to the element comprises:

discovering a pointer associated with a process identifier;

disassociating the process identifier from the pointer;

determining a pointer to a subsequent element; and

associating the process identifier with the newly determined pointer.

7. (Original) An apparatus for storing and retrieving data comprising:

processor capable of executing an instruction sequence;

memory for storing an instruction sequence;

input unit for receiving data;

first output unit for providing data according to a received data request;

one or more ancillary output units for providing data according to a received data request;

instruction sequences stored in the memory including:

data storage module that, when executed by the processor, minimally causes the processor to:

receive data from the input unit;

allocate a data element to accommodate the data;

create a reference to the data element;

store the reference in at least one of a header pointer and a forward pointer included in a preceding data element;

and store the data in the data element;

data service module that, when executed by the processor, minimally causes the processor to:

recognize a data request from the first output unit to the exclusion of all other data requests;

provide data to the first output unit from a data element according to a data element reference and also advance the data element reference to a subsequent data element while a breakpoint is not encountered;

mark a subsequent data element as in-use when a breakpoint is encountered;

create a recommencement reference to a subsequent data element; and

enable recognition of other data requests.

8. (Original) The apparatus of claim 7 wherein the data service module, when executed by the processor, further minimally causes the processor to:

recognize a data request from the first output unit to the exclusion of all other data requests; and

provide data to the first output unit from a data element according to the recommencement reference.

9. (Original) The apparatus of claim 7 wherein the data service module causes the processor to create a recommencement reference by minimally causing the processor to:

retrieve a pointer to a data element subsequent to a current data element;

determine an identifier associated with the data request received from the first output unit; and

store the retrieved pointer and the determined identifier in an associative manner.

10. (Original) The apparatus of claim 7 wherein the data service module causes the processor to mark a subsequent data element as in-use by minimally causing the processor to increment a use counter included in a subsequent data element.

11. (Original) The apparatus of claim 7 wherein the data service module further minimally causes the processor to receive a delete data request from an output unit by minimally

causing the processor to:

determine if a data element to be deleted is in-use;

update a recommencement reference to refer to a data element that is subsequent to the data element to be deleted; and

delete the data element according to the received delete data request.

12. (Original) The apparatus of claim 11 wherein the data service module causes the processor to update a recommencement reference by minimally causing the processor to:

discover a pointer according to a data request identifier; and

replace the pointer with a pointer to a data element that is subsequent to the data element to be deleted.

13. (Original) A computer readable medium having imparted thereon one or more instruction sequences for storing and retrieving data comprising:

data storage module that, when executed by a processor, minimally causes the processor to:

receive data from an input unit; allocate a data element to accommodate the data;

create a reference to the data element;

store the reference in at least one of a header pointer and a forward pointer included in a preceding data element; and

store the data in the data element;

data service module that, when executed by a processor, minimally causes the processor to:

recognize a data request from a first output unit to the exclusion of all other data requests;

provide data to a first output unit from a data element according to a data element reference and also advance the data element reference to a subsequent data element while a breakpoint is not encountered;

mark a subsequent data element as in-use when a breakpoint is encountered;

create a recommencement reference to a subsequent data element; and

enable recognition of other data requests.

14. (Original) The computer readable medium of claim 13 wherein the data service module, when executed by a processor, further minimally causes the processor to:

recognize a data request from a first output unit to the exclusion of all other data requests; and

provide data to a first output unit from a data element according to the recommencement reference.

15. (Original) The computer readable medium of claim 13 wherein the data service module causes a processor to create a recommencement reference by minimally causing the processor to:

retrieve a pointer to a data element subsequent to a current data element;

determine an identifier associated with a data request received from a first output unit; and

store the retrieved pointer and the determined identifier in an associative manner.

16. (Original) The computer readable medium of claim 13 wherein the data service module causes a processor to mark a subsequent data element as in-use by minimally causing the

processor to increment a use counter included in a subsequent data element.

17. (Original) The computer readable medium of claim 13 wherein the data service module further minimally causes the processor to receive a delete data request from an output unit by minimally causing the processor to:

determine if a data element to be deleted is in-use;

update a recommencement reference to refer to a data element that is subsequent to the data element to be deleted; and

delete the data element according to the received delete data request.

18. (Original) The computer readable medium of claim 17 wherein the data service module causes the processor to update a recommencement reference by minimally causing the processor to:

discover a pointer according to a data request identifier; and

replace the pointer with a pointer to a data element that is subsequent to the data element to be deleted.

19. (Original) An apparatus for storing and retrieving data comprising:

means for locking a linked list;

means for retrieving data from an element in the linked list and also advancing to a subsequent element while a breakpoint is not encountered;

means for marking the subsequent element in the linked-list as in-use when a breakpoint is

encountered;

means for creating a recommencement reference to the subsequent element; and

means for unlocking the linked list.

20. (Original) The apparatus of claim 19 further comprising:

means for locking the linked list;

means for determining a subsequent element in the linked list according to the recommencement reference; and

means for retrieving data from the determined subsequent element.

21. (Original) The apparatus of claim 19 further comprising a means for deleting an element in the linked-list.

22. (Currently Amended) The apparatus of claim 21 wherein the means for deleting an element comprises:

means for determining if the element to be deleted is in-use;

means for updating a reference to the element to refer to a subsequent element in the linked list when the element [[in]] is in-use; and

means for deleting the element.